COMPARING TECHNIQUES FOR PROVING UNSATISFIABILITY

Olga Tveretina, Hans Zantema

Department of Computer Science
Technical University of Eindhoven
P.O. Box 513, 5600 MB Eindhoven, The Netherlands
Email: o.tveretina@tue.nl, h.zantema@tue.nl

ABSTRACT

We compare two standard techniques for satisfiability (SAT), which are basic for verification of microprocessor systems. We propose an approach for construction of shorter resolution refutations based on a standard approach called DPLL.

INTRODUCTION

Many problems in different fields, including software verification, electronic design automation, verification and diagnosis of faults in hardware can be naturally encoded into the satisfiability problem for propositional logic(SAT) and then solved by a SAT-solver.

A digital signal processor is a special type of microprocessor chip. The design of increasingly complex digital circuits includes many levels of model transformation starting from high design level where device is presented as a model specification (description) to circuit layout (realization) moving through many intermediate design levels. Each circuit design level has to be verified before production can take place. If it can be proven that an implementation satisfies the given specifications then the chip design is functionally correct (Fig.1).

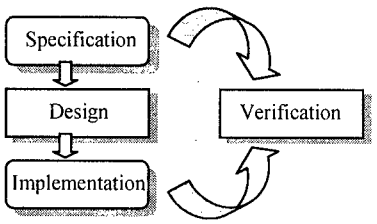

Fig.1. Equivalence checking

Various techniques to formally verify discrete systems have been developed within last decade. These techniques can be roughly divided into theorem proving and model checking. Last ones are restricted to systems with a small number of variables due to a large computational effort.

Methods of propositional logic are applied for formal verification of discrete systems with large number of variables.

In our paper we consider some aspects of formal verification based on propositional formulas validity. This provides a formal mathematical proof that the functional behavior of the specification and the implementation coincides by means of proving of equivalence of two propositional formulas.

It is well-known that the satisfiability problem for the propositional calculus is NP-complete. Roughly speaking, it means that any algorithm to the satifiability problem will have exponential worst case complexity.

PROPOSITIONAL FORMULAS AND SATISFIABILITY

A propositional formula F is a combination of propositional variables $x_1,...,x_n$, where each variable intended to get a value *false* or *true*, in a meaningful way with symbol $\vee$ (or, disjunction), $\wedge$ (and, conjunction), $\neg$ (negation), $\rightarrow$ (implication), $\leftrightarrow$(iff).
Verifying a property P and Q are equivalent means proving that $P \leftrightarrow Q$ is always *true*, which is equivalent to $\neg(P \leftrightarrow Q)$ is not satisfiable.
Here satisfiability means that there is an interpretation for the variables such that the formula becomes *true*, otherwise the formula is unsatisfiable.
We consider methods for formulas in conjunctive normal form (CNF), being a conjunction of clauses, where a clause is a disjunction of literals, and a literal is a variable or the negation of a variable. A particular case of a clause is the empty clause, which can be considered as a representation of *false*. Any formula in propositional logic can be converted to a formula in Conjunctive Normal Form (CNF).

SOME TECHNIQUES FOR PROVING UNSATISFIABILITY

One of the simplest and widely investigated method for proving unsatisfiability of propositional formulas is resolution [5]. Essentially, this method consists of a single rule stating that from knowing $x \vee C$ and $\neg x \vee D$ one may conclude $C \vee D$. An application of this rule is called a resolution step. The particular case of resolution when C or D is empty is called a unit resolution.
A CNF is unsatisfiable if and only if the empty clause can be derived by resolution. Such a sequence of resolution steps ending in the empty clause called a resolution refutation.
The original resolution based algorithm was DP procedure [1]. Unfortunately, it consumes a large amount of space.
A related method is the DPLL procedure [2]. DPLL based procedures are basis for almost all complete SAT solvers, including the fastest ones. This method consists of a combination of unit resolution, doing case analysis upon x and $\neg x$, and going on recursively.

TRANSFORMING DPLL TO RESOLUTION

In [6] we have analyzed the relation between DPLL and resolution in detail. We have proved that if in the arbitrary DPLL procedure s unit resolution steps are executed and r recursive calls are done then there exists a resolution refutation of length at most $s - r/2$.
Adding the restriction that all possible unit resolution steps have to be done after every recursive call we get the stronger upper bound $s - r$.

We have proved for suitable formulas that the second upper bound is tight and no shorter resolution refutation exists than we give by our transformation.
Based on this theoretical result we give a construction of resolution refutation of length less than number of unit resolution steps executed during DPLL.

CONCLUSIONS

Recently there has been interest in using resolution in combination with DPLL search [4]. It is shown that algorithms combining resolution and search are more efficient than DPLL.
We have proved that if in the DPLL procedure s unit resolution steps are executed and r recursive calls are done than the resolution refutation of length at most $s-r$ can be constructed. We implemented this procedure in C. Since DPLL allows freedom of choosing branching variable it is difficult to draw general conclusions from a one particular choice. For examples we have made experiments it turned out that for some formulas had length less than presented upper bound.

REFERENCES

[1] M. Davis, H. Putnam. A computing procedure for quantification theory. Journal of the ACM, 7:201-215, 1960.
[2] M. Davis, G.Logemann, and D.Loveland. A machine program for theorem proving. C.ACM5/1962.
[3] A. Van Gelder. Satisfiability testing with more reasoning and less guessing. In D.S.Johnson and M. Trick, editors, Clique, Coloring, and Satisfiability: Second DIMACS Implementation Challenge, DIMACS Series in Discrete Mathematics and Theoretical Computer Science. American Mathematical Society, 1995.
[4] I. Rish, R. Dechter. Resolution versus search: Two strategies for SAT. IOS Press, 2000.
[5] J. A. Robinson. A machine oriented logic based on resolution principle. JACM12/ 1965.
[6] O. Tveretina, H. Zantema. Transforming DPLL to resolution. Eindhoven technical university. Technical report. To appear. http://www.win.tue.nl/~hzantema/other.html